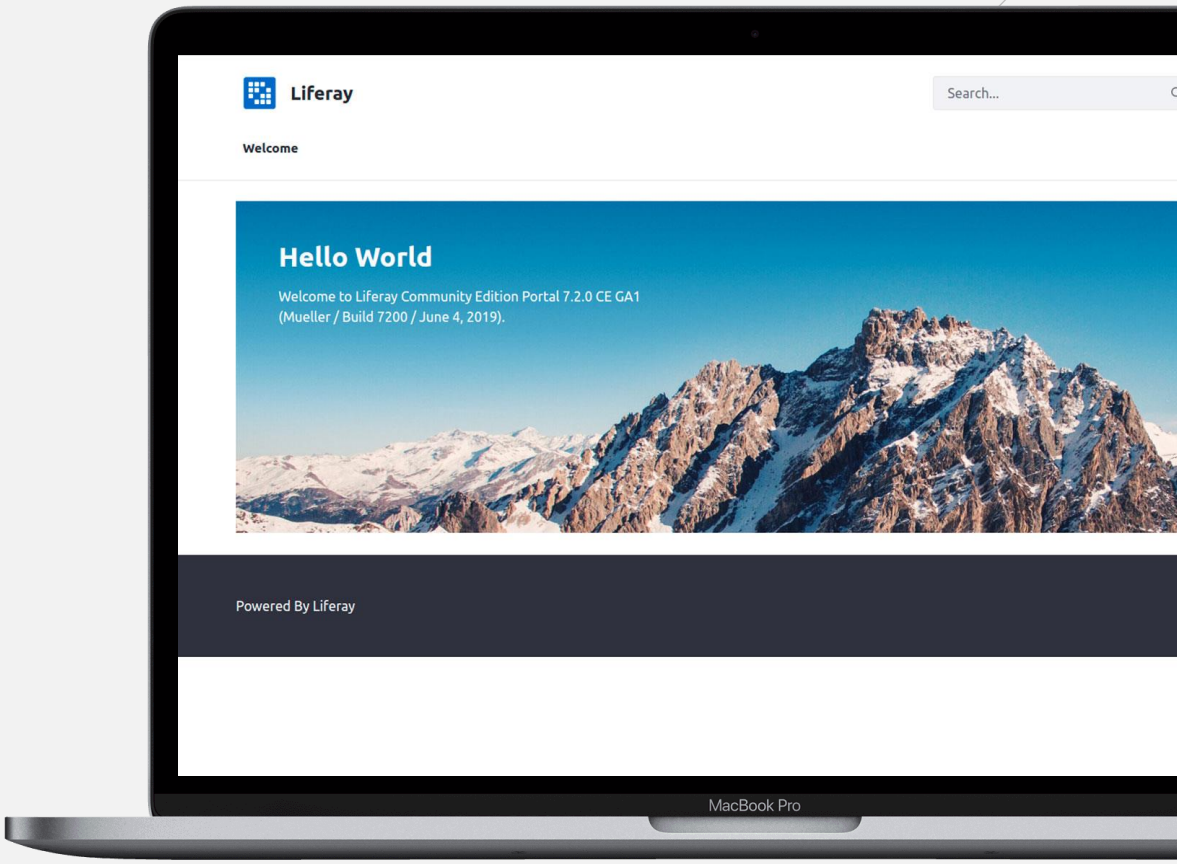
LIFERAY TRAINING

Docker

Agenda

1. What is Docker?
2. Installation
3. Prerequisites
4. Summary



FAST FORWARD IT
Angermunder Straße 126
40489 Düsseldorf
Fon: +49 (0)203 392 333 70

Referent: Chris Börgermann
Datum: 4. Dezember 2020

STADT WUPPERTAL

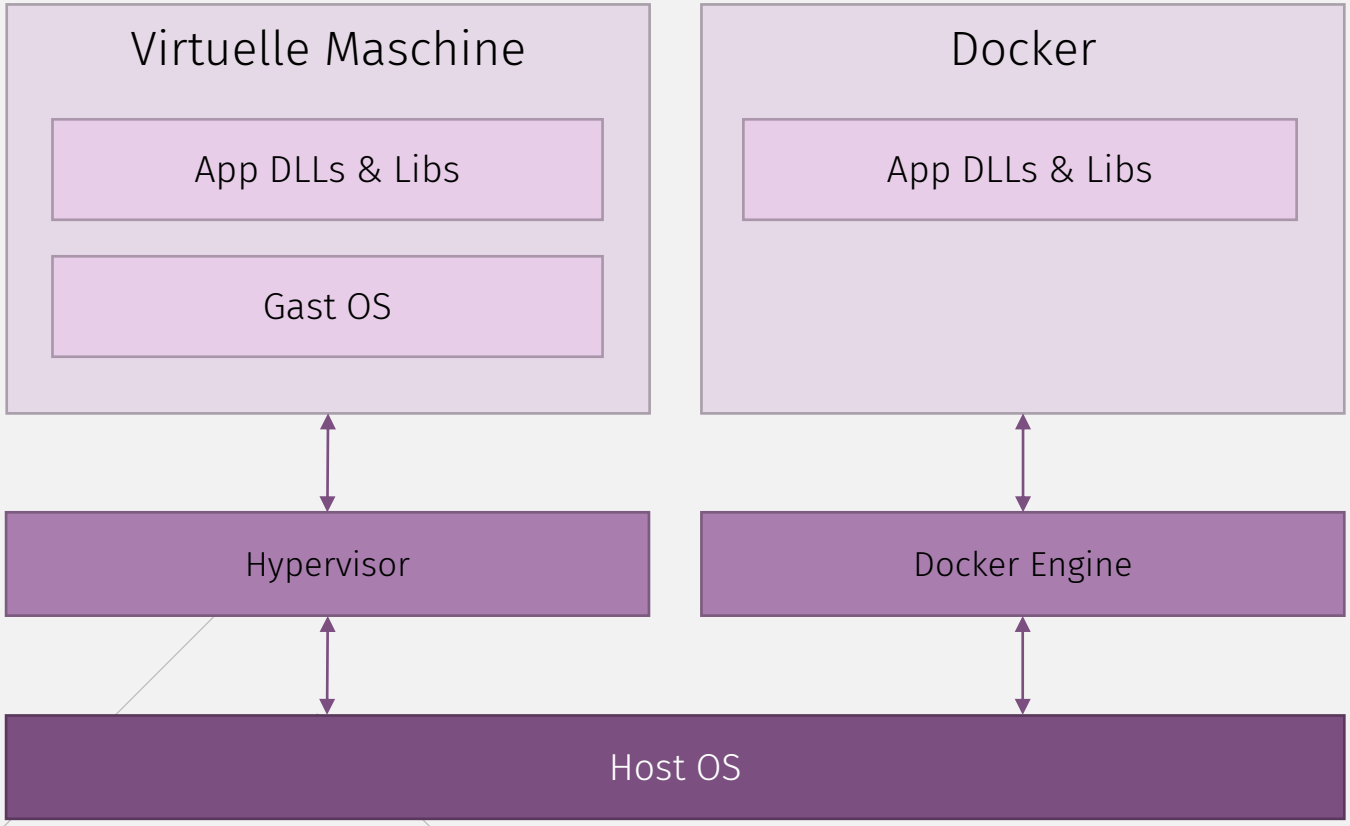
Docker

What is Docker?

- **Virtualization of applications**
- **Independent from**
  - Operating system
  - Virtual machine
- **Container-based technology**
- **Docker Engine**

Docker

Docker vs. virtual machines



Docker

Advantages

- **No operating system required**
- **Portable**
- **Resource-efficient (CPU, RAM)**
- **Memory-efficient (physical storage)**
- **Fast execution**
- **Container run independently**
- **High scalability**

→ *Advantage of virtual machines: Multiple OS can run on one host.*

Docker

Functionality

- **Docker Engine (allocation of resources on hosts)**
- **Docker Compose**
- **Docker file → image**
  - Docker Hub: Repository for Docker images
- **Configuration of the virtualization in docker-compose.yml**
  - Compose environment
  - References (communication between containers)
  - Start container, be happy!

LIFERAY TRAINING

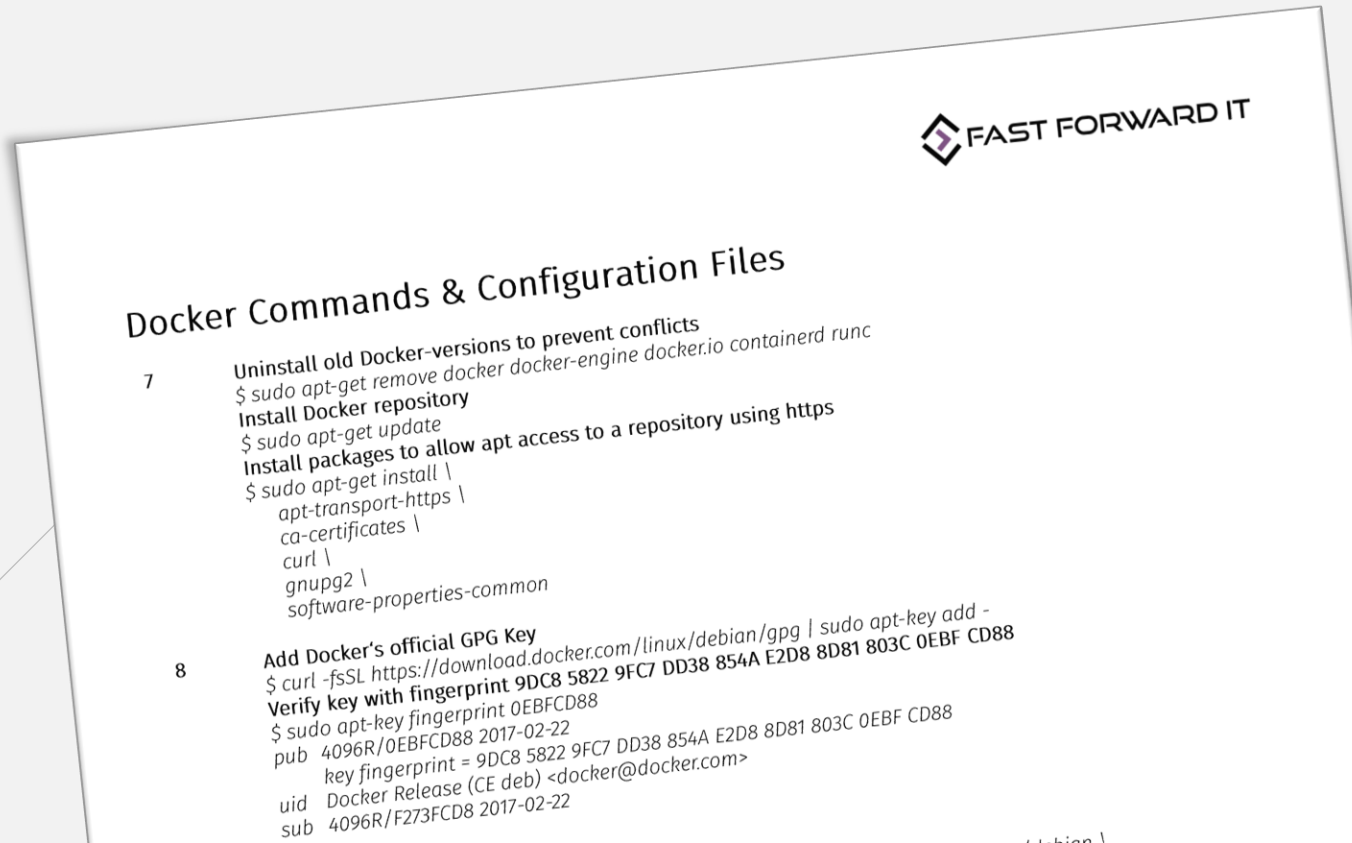
2. Installation

FAST FORWARD IT

Handout

Docker Commands & Configuration Files

bit.ly/DockerCommands



Docker Packages

Installation

- **Uninstall old Docker-versions to prevent conflicts**

```
$ sudo apt-get remove docker docker-engine docker.io containerd runc
```

- **Install Docker repository**
  - Update apt package index

```
$ sudo apt-get update
```

- **Install packages to allow apt access to a repository using https**

```
$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg2 \
    software-properties-common
```

Docker Engine

Integrity validation

- **Add Docker‘s official GPG Key**

```
$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
```

- **Verify key with fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88**

```
$ sudo apt-key fingerprint 0EBFCD88

pub   4096R/0EBFCD88 2017-02-22
      Key fingerprint = 9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
uid   Docker Release (CE deb) <docker@docker.com>
sub   4096R/F273FCD8 2017-02-22
```

Docker Repository

Setup

- **Setup stable repository**
  - To add a nightly or test repository, add the word nightly/test after the word stable
  - Select architecture: replace _____ with
    - X86_64 / amd64 → amd64
    - armhf → armhf
    - arm64 → arm64
    - ppc641e (IBM Power) → ppc64el
    - S390 (IBM Z) → s390x

```
$ sudo add-apt-repository \
        "deb [arch=_____] https://download.docker.com/linux/debian \
        $(lsb_release -cs) \
        stable"
```

Docker Engine

Installation I

- **Install Docker Engine (Community)**
  - Update apt package index

```
$ sudo apt-get update
```

- **Install latest version of Docker Engine (OR)**

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

- **Install specific version**
  - List available versions in your repository

```
$ apt-cache madison docker-ce
```

Docker Engine

Installation II

- **install specific version e.g. 18.06.0~ce~3-0~debian**
  - Update apt package index

```
$ sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING> containerd.io
```

- **Verify Docker Engine**

```
$ sudo docker version
```

- **Update Docker Engine Community**
  - Download update and reinstall Docker Engine

```
$ sudo apt-get update
```

Docker Compose

Installation

→ *Compose is a tool for defining and running multi-container Docker applications*

- **Download current stable release of Docker Compose**

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.25.3/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

- **Apply executable permissions to the binary**

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

- **Test installation**

```
$ docker-compose --version
```

  - If docker-compose fails, check the path! You can create a link to usr/bin or any other directory in your path e.g.

```
$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

LIFERAY TRAINING

3. Prerequisites

FAST FORWARD IT

Docker

First steps

- **Get Docker information**

```
$ docker info
```

- **Start test-container**

```
$ docker run hello-world
```

- **Check if container is running**

```
$ docker info
```

Docker

Datenpersistenz: Volumes – Shared Folder

- **Persistent Storage Volume (PSV)**
  - Storage device or volume
    - Mounted to the host machine directly
    - Mounted as network storage device
    - Cloud storage
  - Mounted under a directory inside Docker Container
  - Container gets exclusive access to PSV
- **Shared Folder**
  - local directory mounted as PSV inside Docker Container
  - data will be shared
  - easy data transfer (CRUD)

Overview

Setup Liferay Environment

- **Configure Docker images**
  - MySQL 8
  - Elastic Search
  - Liferay
- **Script to create images**
  - Create images
- **Create Docker Environment**
  - shared folder
  - setenv.sh
  - .env
  - docker-compose.yml
  - portal-setup-wizard.properties

MySQL 8

Configuration I

- **Create folder/file structure**

  Docker-images

  → Mysql-8

  - Dockerfile
  - **createlportal**.sql
  - my.cnf

- **Dockerfile**

```
FROM mysql:8
COPY my.cnf /etc/mysql/conf.d/custom.cnf
RUN chmod 600 /etc/mysql/conf.d/custom.cnf
ADD createlportal.sql /docker-entrypoint-initdb.d
```

MySQL 8

Configuration II

- **createlportal.sql**

```
CREATE DATABASE lportal;
```

MySQL 8

Configuration III

- **my.cnf**

```
[mysqld]
max_allowed_packet = 1024M
sort_buffer_size = 16M
read_buffer_size = 8M
buffer to avoid disk seeks
read_rnd_buffer_size = 8M
query_cache_limit = 4M
innodb_buffer_pool_size = 2048M
join_buffer_size = 256M
table_open_cache = 400
query_cache_size = 32M
```

Elastic Search

Configuration I

- **Create folder/file structure**

  Docker-images

  → elasticsearch

  → config

  🗋 elasticsearch.yml

  🗋 Dockerfile

- **Dockerfile**

```
FROM elasticsearch:6.5.4
MAINTAINER cb@fast-forward-it.de
ADD config/*.yml /tmp/
RUN cp -P /tmp/*.yml /usr/share/elasticsearch/config
RUN chown -R elasticsearch:elasticsearch /usr/share/elasticsearch/config
RUN /usr/share/elasticsearch/bin/elasticsearch-plugin install analysis-smartcn
RUN /usr/share/elasticsearch/bin/elasticsearch-plugin install analysis-kuromoji
```

Elastic Search

Configuration II

- **Elasticsearch.yml**

```
clusterName: LiferayElasticsearchCluster
network.host: _eth0_
```

Liferay

Folder structure

- **Create folder/file structure:**

Docker-images
→ liferay-7.2.0
→ config
🗋 server.xml
🗋 Dockerfile

Liferay

Configuration I

- **Dockerfile**

```
FROM openjdk:8-jdk
MAINTAINER Chris Börgermann <cb@fast-forward-it.de>
WORKDIR /opt
ENV LIFERAY_HOME=/opt/liferay
ENV LIFERAY_EXTRACT_PATH=/opt/liferay-ce-portal-7.2.0-ga1
ENV CATALINA_HOME=$LIFERAY_HOME/tomcat-9.0.17
ENV PATH=$CATALINA_HOME/bin:$PATH
ENV
LIFERAY_TOMCAT_URL=https://netix.dl.sourceforge.net/project/lportal/Liferay%20
Portal/7.2.0%20GA1/liferay-ce-portal-tomcat-7.2.0-ga1-20190531153709761.7z
```

Liferay

Configuration II

- **Dockerfile**

```
RUN apt-get update && apt-get -y upgrade && apt-get install -y curl ca-
certificates tar p7zip-full && apt-get clean && rm -rf /var/lib/apt/lists/*
RUN curl -s -k -L -C - $LIFERAY_TOMCAT_URL -o liferay.7z
RUN 7za x liferay.7z
RUN rm liferay.7z
RUN mv $LIFERAY_EXTRACT_PATH $LIFERAY_HOME
```

Liferay

Configuration III

- **Dockerfile**

```
ADD config/server.xml $CATALINA_HOME/conf/server.xml
EXPOSE 8080/tcp
EXPOSE 8088/tcp
ENTRYPOINT ["/opt/liferay/tomcat-9.0.17/bin/catalina.sh","run"]
```

Liferay

Configure Docker images I

- **server.xml**

```
<? xml version='1.0' encoding='utf-8‘ ?>
<Server port="8005" shutdown="SHUTDOWN">
<Listener className="org.apache.catalina.startup.VersionLoggerListener" />

<Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />

<Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
<Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
<Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />
```

Liferay

Configure Docker images II

- **server.xml**

```
<GlobalNamingResources>
   <Resource name="UserDatabase" auth="Container"
             type="org.apache.catalina.UserDatabase"
             description="User database that can be updated and saved"
             factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
             pathname="conf/tomcat-users.xml" />
  </GlobalNamingResources>

<Service name="Catalina">

<Executor name="tomcatThreadPool" namePrefix="catalina-exec-", maxThreads="150"
minSpareThreads="4"/>
```

Liferay

Configure Docker images III

- **server.xml**

```
<Connector port="8080" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443"
URIEncoding="UTF-8" />

<Connector port="8088" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443"
URIEncoding="UTF-8" proxyPort="443" scheme="https" secure="true" />

<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" URIEncoding="UTF-8" />

<Engine name="Catalina" defaultHost="localhost">
        <Realm className="org.apache.catalina.realm.LockOutRealm">
                <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
                resourceName="UserDatabase"/>
        </Realm>
```

Liferay

Configure Docker images IV

- **server.xml**

```
<Host name="localhost"  appBase="webapps" unpackWARs="true" autoDeploy="true">

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
prefix="localhost_access_log" suffix=".txt" pattern="%h %l %u %t &amp;quot;%r&amp;quot; %s %b" />

</Host>
</Engine>
</Service>
</Server>
```

Docker

Build images

- **buildDockerimages.sh**

```
cd ../mysql-8
docker build -t ffit/mysql:8 .
cd ../liferay-7.2.0
docker build -t ffit/liferay:7.2.0 .
cd ../elasticsearch
docker build -t ffit/elasticsearch:6.5.4 .
```

→ *Run script to create Docker images*

Docker

Create Docker environment

- **Create folder/file structure:**

  Docker
  - → shared
  - 🗋 docker-compose.yml
  - 🗋 portal-setup-wizard.properties
  - 🗋 .env

Docker

Create Docker environment (MySQL)

- **docker-compose.yml**

```
version: '3'
services:
  mysql:
    ports:
      - "3306:3306"
    environment:
      - MYSQL_ROOT_PASSWORD=PASSWORD
    volumes:
      - ${MYSQL_SHAREDFOLDER}:/var/lib/mysql
      - ${TMP_FOLDER}:/srv
    restart: ${RESTART}
    image: ffit/mysql:8
```

Docker

Create Docker environment (phpMyAdmin)

- **docker-compose.yml**

```
phpmyadmin:
    depends_on: ['mysql']
    ports:
      - "8890:80"
    links:
      - mysql:db
    restart: ${RESTART}
    image: phpmyadmin/phpmyadmin
```

Docker

Create Docker environment (Elastic Search)

- **docker-compose.yml**

```
elastic:
  ports:
    - "9200:9200"
    - "9300:9300"
  environment:
    - ES_JAVA_OPTS=-Xms2G -Xmx2G
  volumes:
    - ${ELASTIC_SHAREDFOLDER}:/usr/share/elasticsearch/data:rw
    - ~/.bash_history:/root/.bash_history
  restart: ${RESTART}
  image: ffit/elasticsearch:6.5.4
```

Docker

Create Docker environment (Liferay I)

- **docker-compose.yml**

```
liferay:
#    entrypoint: "tail -f /dev/null"
    depends_on: ['mysql','elastic']
    ports:
      - "8080:8080"
      - "11311:11311"
      - "8000:8000"
    environment:
      - JAVA_OPTS=${LIFERAY_JAVA_OPTS}
```

Docker

Create Docker environment (Liferay II)

- **docker-compose.yml**

```
volumes:
    - ${LIFERAY_INSTALLATION}/deploy:/opt/liferay/deploy
    - ${LIFERAY_INSTALLATION}/osgi/modules:/opt/liferay/osgi/modules
    - ${LIFERAY_INSTALLATION}/osgi/war:/opt/liferay/osgi/war
    - ${LIFERAY_INSTALLATION}/data:/opt/liferay/data
    - ./portal-setup-wizard.properties:/opt/liferay/portal-setup-wizard.properties
    - ./setenv.sh:/opt/liferay/tomcat-9.0.17/bin/setenv.sh
```

Docker

Create Docker environment (Liferay III)

- **docker-compose.yml**

```
 links:
      - mysql:db
      - elastic
      - mailcatcher
    networks:
      default:
        aliases:
          - liferay
    restart: ${RESTART}
    image: ffit/liferay:7.2.0
networks:
  default:
```

Docker

Setup configuration I

- **portal-setup-wizard.properties**

```
# Upload max 300MB
com.liferay.portal.upload.UploadServletRequestImpl.max.size=314572800

# enable in production!
module.framework.properties.lpkg.index.validator.enabled=false

#timezone
user.timezone=Europe/Berlin

jdbc.default.driverClassName=com.mysql.cj.jdbc.Driver
jdbc.default.password=PASSWORD
jdbc.default.url=jdbc:mysql://db/lportal?useUnicode=true&characterEncoding=UTF-
8&useFastDateParsing=false
jdbc.default.username=root
liferay.home=/opt/liferay
setup.wizard.enabled=false
```

Docker

Setup configuration II

- **portal-setup-wizard.properties**

```
theme.css.fast.load=false
theme.css.fast.load.check.request.parameter=true
theme.images.fast.load=false
theme.images.fast.load.check.request.parameter=true

javascript.fast.load=true
javascript.log.enabled=false

layout.template.cache.enabled=false

browser.launcher.url=
combo.check.timestamp=true
minifier.enabled=false
openoffice.cache.enabled=false
```

Docker

Setup configuration III

- **portal-setup-wizard.properties**

```
com.liferay.portal.servlet.filters.cache.CacheFilter=false
com.liferay.portal.servlet.filters.etag.ETagFilter=false
com.liferay.portal.servlet.filters.header.HeaderFilter=false
com.liferay.portal.servlet.filters.themepreview.ThemePreviewFilter=true

session.timeout=60
session.timeout.warning=0
session.timeout.auto.extend=true
```

Docker

Configuration

- **.env**

```
LIFERAY_INSTALLATION=./shared/liferay

LIFERAY_JAVA_OPTS=-XX:NewSize=700m -XX:MaxNewSize=700m -
XX:+CMSParallelRemarkEnabled -XX:SurvivorRatio=20 -XX:ParallelGCThreads=8
MYSQL_SHAREDFOLDER=./shared/mysql
ELASTIC_SHAREDFOLDER=./shared/elasticsearch
TMP_FOLDER=./shared/tmp

COMPOSE_PROJECT_NAME=FFIT_LR_7-2
RESTART=always
```

Docker

Start Liferay I

- **Open Docker folder in terminal**
- **Run following command**

```
$ docker-compose up –d
```

- **Open browser**

```
https://localhost:8080
```

Docker

Start Liferay II

- **Run following commands (lportal will be populated)**

```
$ docker stop ${docker ps -aq}
$ docker-compose stop
$ docker-compose up -d
```

- **Open browser**

```
https://localhost:8080
```

LIFERAY TRAINING

4. Summary

FAST FORWARD IT

Docker

Most useful commands

- **docker version**
- **docker info**
- **docker images oder docker image ls**
- **docker ps -a**
- **docker run --rm <imageName>**
- **docker exec –it <containerName> <command>**
- **docker stop <containerName>**
- **docker start <containerName>**
- **docker rm <containerName>**
- **docker rmi <imageName>**